(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) Publication number: **0 495 305 A2**

(12) EUROPEAN PATENT APPLICATION

(21) Application number: **91311545.7**

(51) Int. Cl.^{5}: **G06F 15/60**

(22) Date of filing: **11.12.91**

(30) Priority: **14.01.91 US 641031**

(43) Date of publication of application:
22.07.92 Bulletin 92/30

(84) Designated Contracting States:
AT BE CH DE DK ES FR GB IT LI NL SE

(71) Applicant: **VARI-LITE, INC.**
201, Regal Row
Dallas Texas 75247(US)

(72) Inventor: **Taylor, Brooks W.**
P.O. Box 560291
Dallas, Texas 75356(US)
Inventor: **Meldrum, Andrew James Patrick**
902 North Winnetka
Dallas, Texas 75208(US)

(74) Representative: **Read, Matthew Charles et al**
Venner Shipley & Co. 368 City Road
London EC1V 2QA(GB)

(54) Creating and controlling lighting designs.

(57) Disclosed is a modelling and control system for creating lighting designs off-line and for controlling on-line the operation of the actual lighting systems producing those designs. Using a programmable computer, the modelling system prompts inputs by the designer/programmer describing model objects and their parameters including stage dimensions, set pieces and their locations, performers and their positions, trusses, lamps and their parameters such as color, focus, intensity, beam edge, and the relations among the objects. The sytem computes the resutant lighting scenes, organized as cues and arranged and formatted to be compatible with actual lighting controller usage. The system displays the model environment in such a way as to facilitate the modelling operation and its results. It also also generates a variety of spreadsheets for use by the designer.

EP 0 495 305 A2

limited. Consequently, to be truly effective and affordable, the modeling must involve uncomplicated, preferably intuitive tasks; and it must invoke fairly simple instrumentalities which nevertheless are sufficiently revealing or suggestive to enable the designer to visualize the real effect and the states or values of the variables contributing to that effect. Current systems are deficient in these respects.

Ideally, the system should also simulate the console controls to avoid the need for mastering new control/effect relationship.

It is accordingly an object of the invention to provide an improved lighting design instrument which permits off-line creation of lighting designs and thereby reduces dependence on the performers, the theatre and the lighting equipment.

A further object of the invention is to provide an instrument which facilitates the lighting designer's visualization of various lighting effects without the need to access the actual lighting equipment.

Still another object of the invention is to present the user with both schematic and realistic graphical displays, thereby allowing the user to eliminate representations of model data which are not of interest and permitting the use of low resolution or monochrome displays.

It is yet another object of the invention to provide off-line programming of lighting productions which can be simply and rapidly integrated into the actual lighting equipment system.

Yet another object of the invention is to provide an instrument for off-line creation of lighting productions which can be implemented with a relatively inexpensive computer system and which nevertheless computes and displays revealing simulations of lighting parameters.

A still further object of the invention is to provide an off-line lighting show design tool the output of which can be used to directly control lighting instruments. It is also an object to provide the output in a form permitting real time rendering of the lighting design.

Another object is to duplicate the control functions of the automated lighting console, allowing the modelling system to be used to directly control the luminaires in a situation where the effects of operator actions on the luminaires cannot be directly observed by the operator either from the modelling system or console location.

Still another object of the invention is to provide an off-line lighting design programming tool which can symbolize and simulate complex lighting relationships in a simple, uncluttered and revealing manner and without excessive response time.

It is also an object of the invention to provide such a programming instrument wherein there is produced a variety of displays suited to the varying needs of the designer.

Another object of the invention is to provide an off-line modelling system for a lighting set in which prompts for the user bear an analogous relationship to actual console configurations.

Yet another object of the invention is to provide an off-line modelling instrument for lighting productions which offers the user a choice of methods in specifying model data, and in which menus and dialog boxes facilitate use of the instrument. By embedding in the system terms and notations familiar to the lighting designer, and producing the customary actions associated with them, the user is able to communicate with the system using the same vocabulary and achieving the same results as would be the case with a human rather than machine lighting designer assistant.

Further objects of the invention are to provide an off-line modelling system for lighting designs in which a mathematically correct three dimensional model of the performers, the set, the trussing and the luminaires of the show is computed and displayed; in which pan and tilt values can be calculated in terms of target position; in which model elements are automatically adjusted when related elements are changed, and in which updating and revising is simply accomplished.

SUMMARY OF THE INVENTION

In the implementation of the invention, a system is provided for modelling a lighting production including the characteristics of the production site and the lighting scenes produced by the lighting console and the lighting instruments without requiring access to the site and instruments. The system utilizes a computer having data entry means, a processor, storage means, a monitor and a graphics generating program. The modelling system further includes:

(1) means for receiving data:

(a) describing relevant parameters of the lighting site including lighting targets;

(b) identifying the selection of the lighting instruments to be operational in the production of the lighting designs;

(c) describing the location of the selected lighting instruments;

(d) describing values of the lighting parameters associated with each of said selected lighting

Figures 4a and 4b are views of model data displayed in a spreadsheet format;
Figures 5a and 5b illustrate other spreadsheet formats for the model data;
Figures 6, 7, and 8 illustrate the monitor screen and various manipulation dialogs;
Figure 9 is a view of the monitor screen showing the tools palette;
Figures 10A through 10C are views of the monitor screens showing various object selections;
Figures 11A through 11C are views of the monitor display illustrating other selection activities;
Figures 12 and 13 are diagramatic views of a luminaire illustrating the coordinate axes and angle specifying conventions;
Figures 14 through 17 and Figures 18A, 18B and 18C describe processing steps used in computing various lamp positions;
Figure 19 shows the monitor display screen with the cue store and recall palette; and
Figure 20 shows the monitor display with an actual lighting scene simulated thereon.

DESCRIPTION OF THE PREFERRED EMBODIMENT

Certain aspects of the invention are illustrated in Figure 1 where a computing system 20 is displayed for modelling the lighting production eventually to be produced by the actual lighting system 100. The latter includes a lighting console 110 such as the ARTISAN™ or MINI-ARTISAN™ marketed by Vari-Lite, Inc., and a network of lamps which may for example be various combinations of the models VL2, VL2B, VL3 and VL4 previously referenced.

The lighting controller may be of the type associated with the Vari-Lite VL200 series lighting system and portions of the controller may also be configured as described in United States Patent No. 4,980,806, issued to Taylor and Walsh on December 25, 1990.

The computing system 20 is a general purpose computer running under the graphic system described hereinafter. The preferred computer model is a Apple Macintosh II fx. It includes a microprocessor, RAM, ROM, storage and interfaces with the user via mouse 28 and keyboard 30. A modem for communication is also provided and data can be printed on the hard copy printer.

To facilitate an understanding of the implementation of the invention, certain general factors of the illustrated embodiments are first described. The design includes a modelling system and a method for modelling, programming, and simulating the functions of an automated lighting system such as 100 having variable multi-parameter lighting features. The modelling system and method may be used to write lighting cues in advance.

Using the general purpose computer 20 embodying the graphics generating system and software the programmer/designer can develop a working model of a lighting system configuration and its surrounding environment. The programmer defines the spatial environment, e.g. the theatre or arena, in which the lighting equipment will be used. In addition, the programmer defines each of the elements that comprise the lighting system, stage environment and targets to be illuminated. Using the modelling software to generate a graphical display of the lighting system on monitor 24, the system is capable of creating and manipulating the defined objects.

With the modelling system of the present invention, all that is required to develop the model is knowledge of the dimensions and characteristics of the arena and stage environments and a determination of the number, type, and placement of the lights to be utilized. This can be done in advance without entering the arena or hanging any lights.

Once the programmer has developed the basic model by defining the positional relationship of each of these elements with respect to other elements in a universal frame of reference, the modelling system provides for the graphical creation and manipulation of show objects and their associated attributes in three dimensions on the two dimensional display such as screen 24.

The programmer defines positional and characteristic information for each element in the model by communicating interactively with the modelling system using the mouse 28, keyboard 30, cursor 34 and other screen objects including icons 35, menus 36 and model objects 37.

Cue parameter data may be written by identifying the lights to be assigned, their shape and color, and the position of the associated targets to be illuminated. The cue parameter data may include data representing other desired beam characteristics.

After the programmer has obtained the desired lighting effect by defining the beam characteristics and associating the necessary lights with their corresponding targets, the system can compute, if necessary, the required pan and tilt values for each light so that it is focussed on the desired target. The programmer can define the pan and tilt values either by specifying absolute values for these parameters (thus, not requiring a pan and tilt computation) or he can associate the lights with their targets. When a light is associated with

communicated and reflected in the console and vice-versa.

In the third alternative mode of interaction, the modelling system actually replaces the conventional control console in the control of cue set-up, store and recall. All of the functions of the console are then performed by the modelling system.

Alternatively, the functions of the modelling system can be performed by the control console itself. In the fourth mode of interaction, the modelling system replaces the lights in the actual system, responding to the controls of the actual console as the lights would. The effect of the user's manipulation of the console controls is reflected in the display of the modelling system.

GRAPHICAL VIEW OF THE MODEL

The modelling system of the present invention includes a graphical user interface to interact with the programmer. In general, the preferred interface follows the guideleines designated in Human Interface Guidelines: The Apple Desktop Interface, ISBN 0-201-17753-6. The interface presents the programmer with a graphical view of the model that is a clear, simple, and uncluttered representation of the current state of the objects in the model.

Lighting instruments or other objects can be created, modified and deleted in the graphical view of the model. The model objects can be displayed utilizing symbols such as "spheres" for the luminaires, blocks for the stage performers, and "rods" for the light beams and trusses. These tasks are performed by the programmer, preferably by means of executing menu-driven commands. These functions will be addressed in greater detail below.

The modelling is carried out with the aid of a programmable general purpose computer such as shown in Figure 1. A preferred model is the Apple Macintosh II fx. Computer 20 houses the usual complement of a processor, ROM memory, drives, busses and control circuits and the modelling system.

A typical graphical input device 28 such as a mouse, joystick, tracker ball, or light pen, together with the menu system and cursor 34, allows the programmer to interact with computer 20. Mouse 28 is the preferred graphical input device for pointing to information on display monitor 24. Alphanumeric keyboard 30 provides the usual functions and also serves as an alternative data entry means.

In the preferred embodiment of the present invention, menus are utilized to allow the programmer to efficiently choose an item to be incorporated or updated in the model, or to alter the operating mode of the system. The menus are also utilized to present to the programmer only the legitimate alternatives available, thereby precluding invalid choices. The menus also ameliorate the need to remember command names and give the programmer an overview of all of the alternatives.

Fig. 2a shows a typical graphical display of a lighting system on display monitor 24. A "menu bar" 38 lists the names of typical pull-down menus 44 a - g, from which the programmer may select commands as shown in Figs. 2a -2g. When the programmer selects a menu to be displayed from the "menu bar" 38, alternative commands are presented for further selection. The selection process and the individual pull-down menus 42 a- g are discussed in greater detail below.

As the programmer builds the graphical model that represents the lighting system and surrounding environment a corresponding master data base is created and stored in the memory of computer 20. The master data base stores the parameters and attributes that define the objects that are represented in the model. As the programmer adds an object to the model, a record for that object is created and stored in the master data base. As discussed below, each record has a field for each of the attributes of the object that need to be defined during the modelling and programming process.

The image that is seen on display 24 as the programmer builds the model through an interactive graphics process is stored in the underlying master data base in digital memory as a matrix of values. The screen image is stored in a frame buffer as a pattern of binary numbers representing an array of picture elements (pixels). As changes are graphically made on display 24, the records created in the digital memory are modified to represent the current state of the graphical model.

As the programmer models and programs the lighting system, monitor 24 displays a view of the lighting system and stage environment and shows the programmer the results of adjustments to the lighting parameters. The programmer can observe the effects of changes in color, pan, tilt, beam size, edge, and intensity on the overall lighting effect. Other effects, e.g., gobo selection, are also represented in a similar manner.

Fig. 3 illustrates the display on screen 24 of windows 54 a-c, which are plan, front and side elevation views respective, of the model. In one preferred embodiment, the display utilizes up to four windows to give the programmer the ability to observe the lighting effects from these views, and a perspective view from another selected point (not shown).

modelling file. As the programmer adds elements to the model, the associated data will be included in the various spreadsheet views discussed below.

A first spreadsheet view, termed a "model window", is shown in Fig. 5a. It provides a view of the model objects and their current attributes which allows the programmer to view data that is necessary for the modelling environment, but is not represented in the cue data transferred to the lighting system. For example, in order to develop the model it is necessary to know the dimensions of all of the set pieces. However, this information is not required to perform the actual show. The spreadsheet in the "model window" can be arranged so that the user-assigned names and model types (lights, set pieces, etc.) appear in the row headings, and the names of the attributes of these objects appear in the column headings, as shown in the Figure.

A second type of spreadsheet called a "luminaire window" is shown in Fig. 5b. This spreadsheet presents the programmer with the lighting system data that he is accustomed to programming. Timing data is displayed along with flags or indicators indicating to which parameters the timing applies. This spreadsheet can be arranged so that the channel numbers or other identification of the lights are placed in the row headings and the names of the attributes of these lights are placed in the column headings. This spreadsheet represents the state of the luminaire objects in the model, and the data presented will change as cues are recalled into the lights, as well as when individual lights are manipulated.

A third type of spreadsheet called a "cue window" displays the data stored in a selected cue, regardless of the state of the luminaire objects in the model. This spreadsheet can be arranged so that the channel number or other identifier of the lights appears as row headings and the column headings contain the names of the attributes of the lights. The cue number and additional data stored with the cue related to console functionality are also displayed. The programmer may select whether the attributes of all the lights are shown or only those lights that are active in the selected cue. This spreadsheet can be switched to a "tracking" view of the cue data, presenting the viewer with only the cue data which has changed value from the previous cue (or from any arbitrary cue).

The rows of any spreadsheet can be sorted in a number of ways. The rows of the "luminaire" and "cue" windows, comprised of luminaires or dimmer channels, can be sorted by their channel number. Alternatively, the rows can be arranged so that all of the channels of similar luminaire types (VL2B™, VL3™, VL4™, etc.) can be grouped together, and then can be secondarily sorted by channel number.

The row headers of the "model window" contain the symbolic names of all of the objects in the model. This spreadsheet can be sorted alphabetically by object name for non-luminaire objects and by channel number for the luminaires. Alternatively, the rows can be grouped by object type (e.g., target, luminaire, etc.) and secondarily sorted by name.

The program design also permits the programmer to establish spreadsheet templates that may be used to reformat a spreadsheet or utilize previously established spreadsheet formats. A programmer can create a spreadsheet template by placing a spreadsheet window in the front window on display screen 24 and executing a menu command to create a template of that spreadsheet's format. The menu command invokes a dialogue message that allows the programmer to attach a symbolic name to the template. To store a template, a document is created in memory and written to disk that contains the template name, the spreadsheet type (model, luminaire, or cue window), the object types displayed in the spreadsheet (row headings), their associated attributes (column headings), and the order of these attributes.

Once the programmer has established a desired template he is able to call up a spreadsheet to the front window of display 24 and execute a menu command that is available to reformat the spreadsheet according to that template (or another if he wishes). Upon execution of the menu command, the programmer is presented with a dialogue message containing a listing of all of the available spreadsheet templates that are appropriate for that type of spreadsheet (e.g., model, luminaire, or cue window). A selection can then be made.

The programmer has the means to open and close any of the spreadsheet windows. Additionally, when the information contained in a spreadsheet view exceeds the available display area, the scroll bars on display 24 are enabled that allow the programmer to show portions of the document not currently displayed.

Objects can be created, modified and deleted within the spreadsheet view of the model. Object attributes can be changed in a spreadsheet view with the same menu commands and dialogues used in the graphical view or by selecting one or more cells and entering a new value at the keyboard. As in other modes, new values entered from the keyboard are automatically checked by the system for errors. Almost all of the attributes of the modeled objects can be modified in this manner.

The objects that the programmer wishes to alter in the spreadsheet view must be activated or selected before they can be operated upon. An object can be selected in a spreadsheet window by clicking on the appropriate row header describing the object. A group of objects can be selected in the spreadsheet view

When initially building the model, the programmer can be presented with a menu or listing of building blocks or symbolic representations for all of the elements that would likely be placed in the lighting system model. This building block menu, similar to a legend or key, is presented as a separate screen, a pull-down menu, or a window on a portion of the screen. As shown in Figure 3, when presented with this menu of building block alternatives, the programmer can designate his building block choice to computer 20 by manipulating mouse 28 so cursor 34 points to his selection from among alternatives 66 a-g. Selection of this building block is by depressing key or button 29 provided on mouse 28. The selected "icon" will be distinguished from non-selected items. If a pull-down menu system is utilized to access the building block menu, the programmer must first access the pull-down menu by selecting the menu from menu bar 38. In the preferred embodiment, the building block menu (or "objects palette") is accessed by selecting menu 44d in Fig. 2d. Thereafter the programmer "drags" the cursor down to highlight the "show objects palette" option. The building block menu then appears in a window on the screen.

Once the programmer has accessed the list of building blocks he can begin selecting the desired objects to be included in the model. A typical legend menu 62 is shown in window 58 of Fig. 3. The legend screen or menu 62 comprises symbolic representations 66 a-g or "icons" that are small pictures that represent available elements that can be used in a typical lighting system. The modelling system can be programmed with a directory of the symbolic representations of elements that are common in lighting systems and the surrounding environments, this directory being stored in computer 20. The directory is accessible to the user so that it can be updated by adding or deleting elements as necessary.

Upon the selection of a building block element by the programmer, a record is created in the underlying data base in the storage means of computer 20 to store all of the necessary information for that object. Preferably, the records are broken down into fields for each attribute of the object that is to be stored. As will be discussed below, when the programmer selects an icon the modelling system will know what that icon is, and it can create a record in the memory specifically designed for that type of object. The created record will have fields for every attribute that must be known in order to completely model and program each object. The record creation will be discussed with respect to each building block element below. The information that is created in the memory records during the creation of the model helps to facilitate the cue writing process.

LIGHTING INSTRUMENTS

Icons 66 a-d in the building block menu of Fig. 3 are symbolic representations of various lighting instruments that can be utilized in a lighting system. In a preferred embodiment of the invention, each type of unique lighting instrument is assigned its own symbolic representation. In the legend menu 62 of Fig. 3, each of the elements 66 a-d represent different lighting instruments.

For example, icon 66a represents a standard conventional light that is focussed by hand and whose intensity is controlled by an external dimmer. If the programmer clicks on icon 66a to select a conventional light for inclusion in the model, a record will be created in the underlying data base with appropriate fields and the programmer will be prompted to enter the information that is necessary to define the conventional light. Upon selection of a conventional light, the programmer is prompted to enter the channel number identifying the light, the symbolic name of the support from which the conventional light will hang and the positioning of the light relative to its support.

This information is stored in the appropriate fields of the record associated with this modelled object. Since conventional lights only offer variable intensity control, a field is created in the record to store the intensity value. The record created upon selection of a conventional light can also include such fields as channel number, object type, supporting object, translation (x,y,z translation with respect to its support), rotation (x,y,z rotation with respect to its support), focus (pan and tilt values that will not change with the recall of cues), color (value won't change with the recall of cues) and intensity (representation of the value of the control signal (0 - 100%) that is sent to the dimmer device to which the luminaire is connected). All of the fields except for intensity are generally defined once the programmer has included and defined the light in the model.

Automated lights can also be assigned unique symbolic representations such as the icons 66 b-d, Fig. 3. As in the case of a conventional light, the selection of an icon 66b - 66d (representing an automated light) establishes a record in computer storage with the appropriate fields and the programmer can be prompted to enter channel number, support, and position relative to the support associated with the light for entry into the associated fields.

In one embodiment, icon 66b represents a Vari-Lite "VL2B™" model; 66c represents a Vari-Lite "VL3™" model, and 66d represents a Vari-Lite "VL4™" model. Since the modelling system is pro-

the lighting system. Upon selection of a building block element by clicking on it the programmer is prompted to supply the information needed to define the building block element.

Alternatively, the programmer can select a building block by entering an alphanumeric character on keyboard 30 that is associated with that element.

As information is retrieved from the programmer it is placed in the appropriate fields of the record associated with the selected model object.

An efficient method of prompting the programmer to retrieve necessary definitional information is by means of "dialogue boxes". Typical dialogue boxes are shown in Figs. 6 - 8. The dialogue box is a window that can be used to contain text or symbolic representations.

Once the programmer selects a building block element from legend menu 62, the modelling system will recognize the type of object selected (e.g., truss, target, VL4™, etc.) and can retrieve from memory any previously-stored characteristics or other information describing that object. This minimizes the information that must be entered by the programmer and maximizes the operating efficiency.

2. Assigning a Symbolic Name

The programmer can also use the dialogue box or other data entry means such as keyboard 30 to attach a symbolic name to the graphical object that he has selected. During the programming process the symbolic names provide the programmer with an easy means for identifying particular targets and elements within the lighting system. For changing a defined symbolic name, the programmer can select icon 135 of Figs. 7-9, causing a dialogue box to be presented on the screen. The dialogue box will show the programmer the symbolic name that is presently associated with the selected object and allow it to be updated. The "name" icon 135 can be accessed by means of a pull-down menu or by a palette of tools presented in a window on a portion of the display as shown in Fig. 9.

3. Defining Dimensions

After selection of a building block, the programmer is prompted to enter its dimensions. In the preferred embodiment, the dimensions of the lighting instrument building blocks shown as 66 a - d of Fig. 3 are pre-programmed and the programmer will not be prompted for their dimensions. However, when the programmer selects any of building block icons 66 e - g he will be prompted for the dimensions of the selected object. The entered information is placed in the appropriate fields in the record created for the model object.

In the preferred embodiment, the programmer can update previously defined dimensions of a selected model object by selecting the "dimension icon" 137 of Fig. 9. Upon such selection, a dialogue box will be presented with the dimensions of the selected object. The programmer will be able to update the presented information.

4. Defining Supporting Object

In the preferred embodiment, the programmer is provided with a "support" icon 139 for updating a selected graphical object. Upon selecting a graphical object and clicking on the "support" icon 139, a dialogue box is presented containing the symbolic name of the object that supports the selected object. Thereafter, the programmer will be able to change the supporting object. The "support" icon 139 can be accessed by means of a pull-down menu or from a palette of tools 130 presented in a window on a portion of the screen.

5. Defining Positional Information

In addition to identifying and defining the characteristics for the building block object to be included in the model, the programmer also defines the position of the object within the model. After the programmer has entered all of the definitional information that has been requested, the object is initially inserted in the model in a default location corresponding to the center of the space that supports the object. For example, if the object is a target (e.g. a performer) supported by the stage, the target is placed at the center point of the stage and at zero rotation with respect to the stage. Thus, the graphical object initially has a translation of (0,0,0) and a rotation of (0,0,0) with respect to its support.

The programmer changes the position of the selected object by clicking on its image, moving the objects outline with mouse 28 to the location where the programmer would like to position the object. He

$$[x_2 \quad y_2 \quad z_2 \quad 1] = [x_1 \quad y_1 \quad z_1 \quad 1] \quad \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

Rotation about the y axis is given by:

$$[x_2 \quad y_2 \quad z_2 \quad 1] = [x_1 \quad y_1 \quad z_1 \quad 1] \quad \begin{bmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

And finally, a rotation of angle θ about the z coordinate axis is represented by:

$$[x_2 \quad y_2 \quad z_2 \quad 1] = [x_1 \quad y_1 \quad z_1 \quad 1] \quad \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

By utilizing a concatenation process, any number of successive transformations can be represented by a single transformation matrix. If a first translation is represented by matrix T_1, then a rotation about the x axis represented by matrix R_x, and finally another translation represented by matrix T_2, the concatenated transformation to represent this sequence that places the new coordinates at (x', y', z') is: [x' y' z' 1] = [x y z 1] ($T_1 \cdot R_x \cdot T_2$). The order of operations must be maintained when the transformation matrices are cross multiplied together.

Cross multiplying by the inverse matrix M^I of a matrix M performs the symmetrically opposite transformation. The result of this operation is to undo the effect of the transformation with the matrix M.

Illustration

To illustrate the addition of a building block to the model, the selection of the stage's building block element for inclusion in the model and the definition of its necessary characteristic information is discussed with respect to the preferred embodiment. The stage is generally one of the first elements to be defined and placed in the model. The stage is often the support for numerous objects in the model so many other elements are often defined with respect to the stage location. Referring now to Fig. 3, and as discussed in general above, the programmer selects the symbolic representation of the stage icon 66e by manipulating cursor 34. Upon selection of stage icon 66e, the processing means of computer 20 recognizes that the programmer has selected the symbolic representation for the stage or related platforms. The appropriate record and corresponding fields (as discussed above) is established in the data base stored in computer 20. The programmer's selection of the stage element from the legend menu triggers a request, generally by means of a dialogue box (as shown in Fig. 8), to attach a symbolic name such as "stage" to the object. The symbolic name facilitates later programming, and allows the programmer to identify the stage when using it as a support for other objects.

The processing means of computer 20 recognizes that the programmer has selected icon 66e. The selection of this icon invokes a dialogue message such as that shown in Figure 7, that retrieves the length,

and drags the cursor to include all of the desired lights within the highlighting box 50. The ability to select a predefined group of lights is similar to the "Group Select" option that is a common function of the Vari-Lite control consoles. This feature simplifies the processes performed by the programmer when he wishes to assign the same value for a selected attribute to a group of lights, for example directing a number of lights to the same target. Thereafter, the programmer can assign a symbolic name to the group of lights just selected.

Defining Target Parameters (Pan and Tilt)

The programmer moves the lights to focus on a target by controlling the focus (or pan and tilt) parameter of the lights. The programmer can establish the pan and tilt values of any light in either a direct or indirect manner.

The information recorded in the model has established the position of every element within free space. This information allows the calculation of the pan and tilt values required to point a particular light at a certain target. The pan and tilt values can be calculated by retrieving the location of the particular light and the particular target in free space. This has an advantage over methods of programming employed with the actual lighting system where the pan and tilt values have to be manually adjusted until the light is focused correctly on its target.

The programmer can specify that the pan and tilt values of any light shall be such that the light is focused on a defined target object. Using this "parametric" definition of the pan and tilt values which associates the light or lights with a particular target, the modelling system can compute the appropriate values. To focus the lights in this manner, the programmer indicates the symbolic name of the appropriate target to the computer. The associated target for a light can be retrieved by the programmer by using the dialogue box query of the target's symbolic name. In this embodiment, the programmer indicates to the modelling system the target that the light should be focussed on. Thereafter, the required pan and tilt values are computed. If the resulting tilt is impossible, the programmer is notified by the modelling system.

If a light is focussed by associating a target, the symbolic name of the target and the calculated pan and tilt values can be stored in the light's record. If the target is later moved in the model, the pan and tilt values will be correspondingly changed and updated to maintain focus on the target. Again, if the resulting tilt is impossible, the programmer is notified. When the cue data is transferred to the control console of the lighting system, the pan and tilt values for lights focussed on targets will be transferred as absolute pan and tilt values.

A second method defines the pan and tilt values for the selected lights by entering "absolute" numerical values. This is done by selecting the light or lights to be defined and then responding to a dialogue box query to enter the pan and tilt values.

A third method of defining pan and tilt values for a light is by "preset focus". This method, as implemented in the Vari-Lite console, focusses the lights to their targets by a dereferenced or indirect specification. The programmer selects an arbitrary number of lights, assigns a target to them or enters absolute numerical values and then stores this configuration as a preset focus by attaching an alphanumeric symbolic name to it which serves to identify it.

The preset focus position of all presently activated lights can thus be stored in a file and retrieved at any time by entering the identifying name. This retrieves for each light the pan and tilt values required to focus the light on the desired position.

When a programmer fixes the pan and tilt values for a cue by means of a preset focus, the symbolic name of the preset focus is stored in the cue. The file defining each preset focus is also transferred to the control console. When the data is transferred, the pan and tilt values defined in this manner will be identified by the associated preset focus symbolic name. When a cue using preset focus is executed in a light, the light retrieves the proper pan and tilt value by referring to its preset focus record. Prompts for preset focuses can be presented to the programmer as a graphical duplicate of the preset focus panel on the Vari-Lite console.

A lighting designer uses several different combinations of beam spread and tilt angle to light a performer standing on stage. A 'head shot' uses a narrow beam of light which only illuminates the performer from the neck up to the top of the head. A 'head-and-shoulders shot' is similarly from the shoulders up. A waist shot' illuminates the performer from the waist up. A 'full-body shot' illuminates the performer from the feet to head. For any given luminaire illuminating a performer, the tilt value and beam size (beam angle) will be different for each of these 'shots'.

By embedding knowledge of the proportional sizes of the various parts of the human body in the model, the system can calculate the pan/tilt and beam size for each luminaire to achieve a head shot, waist shot,

The flow chart in Fig. 16 shows a method for determining which pair of azimuth and elevation values will point the light to the target with minimum yoke movement. The azimuth and elevation (calculated in radians) are converted into degrees in step 192. Additionally, it is necessary to determine the corresponding pan-azimuth value (panaz) associated with the present pan setting (pannow) 192. "Panaz" is used in comparing the two pan-azimuths which point to the target and in determining which is preferred.

In step 194 the two possible pan-azimuths that point at the target are obtained by adding 90° to the azimuth (paz1) and subtracting 90° from the azimuth (paz2). The 90° is added or subtracted from the calculated azimuth value so that the plane of tilt rotation will pass through the target. Since a pan-azimuth of 0° is defined in Fig. 12 to be pointing along the x axis, a pan-azimuth of 0° puts the tilt plane in the plane of the y-z axes. In this position, the lamp could point to an azimuth of 90° or 270°. Thus, in order to produce a pan-azimuth from an azimuth, 90° must be added to or subtracted from the desired azimuth.

Since the range of valid pan-azimuths is limited to 0° through 360°, steps 196 - 202 guarantee that "paz1" and "paz2" are maintained in this range (modulo 360°). If paz1 is greater than 360° then 360° will be subtracted from the value (step 198). For example, a pan-azimuth of 400° is beyond the range of the lamp so the value is reduced by 360° to an equivalent pan-azimuth of 40°. Similarly, if paz2 is less than 0° then 360° will be added to it, to keep the pan-azimuth in range (step 202).

As a result of pan stop 150, pan-azimuths of 0° and 360° represent different positions of the yoke, each being on opposite sides of pan stop 150. In steps 204 through 220, it is determined whether the two possible pan-azimuths (paz1 and paz2) are set to either 0° or 360°. If paz1 is equal to 360° and panaz (the present pan-azimuth setting) is less than 180° (step 216) then paz1 should be set to 0° (step 210) in order to minimize the rotation of the lamp. However, if panaz is equal to 180° (step 208) then each direction of rotation is the same, and a "tie-breaker" should determine whether 0° or 360° is used (step 212). This tie-breaker can be under user control. Steps 214 through 220 show the steps for analyzing paz2. After these steps have been performed, there are two possible solutions for pan-azimuth that are 180° apart. These two values are the output of the flow chart shown in Fig. 16.

The flow chart of Fig. 17 (see also Figs. 18a, b and c) determines which of the two solutions should be utilized. Step 230 determines the delta angular distance that must be traveled by the lamp in moving from the present setting to the new setting for each of the two pan-azimuth values (resulting in delta1 and delta2). In step 232, it is determined whether both paths travel the same angular distance. If they do, then the tie-breaker determines which pan-azimuth will be used (step 234).

However, if the paths are not the same, then a determination is made as to which value represents the shortest angular distance (which is the smaller of delta1 and delta2). If the angular distance offered by the first value (paz1) is smaller than the angular distance offered by the second (paz2), then the algorithm proceeds to the branch of the flow chart with steps 240, 246 and 248. Step 240 determines whether you can get to the new pan-azimuth value from the previous pan-azimuth value without traveling through pan stop 150. If so, then the first pan-azimuth value (paz1) is selected and converted to a pan value. The corresponding tilt value is chosen based on the "tilt sense" value, where:

tilt = elevation * tilt sense. (step 250).

If step 240 determines that the light cannot travel from the previous pan-azimuth value to the new pan-azimuth value without going through the pan stop 150, then the longer distance (paz2) should be chosen. Steps 238, 242 and 244 proceed from step 236 when the angular distance to be traveled is shorter for the paz2 value. These steps are evaluated in the same manner as steps 240, 246 and 248.

Since tilt values are limited to the range of +135° and -135°, if an impossible tilt value results then there will be no solution and an error condition will arise.

Example

The pan and tilt computation is discussed below with respect to the position of a target in a lamp's coordinate system. In order to illustrate, a target having coordinates of (20,20,45) in the lamp's coordinate system is discussed. Proceeding initially to the steps shown in Fig. 14, the x and y coordinates of the target are both positive (condition of Step 160), therefore the calculation results in an Azimuth equal to the inverse tangent (arctan) of (20/20) or the arctan of 1. This results in an azimuth value of $\pi/4$ radians (or 45 degrees).

Thereafter, the elevation is calculated according to the method shown in Fig. 15. Since the error condition of Step 180 is not satisfied, the following elevation computation is performed in Step 184:

$$\text{Elevation} = \text{Arccos}\,(45/(20^2 + 20^2 + 45^2)^{1/2}).$$

lighting system (such as color, beam size, intensity, edge and gobo) in a number of ways. One alternative is to define the parameters in an "absolute" manner. As an illustration, if the programmer wishes to define the intensity of the lights in an "absolute" manner, he can specify that all of the selected lights should have an absolute intensity value, for example an intensity of 65%. Parameter data that has been defined in an "absolute" manner will not be automatically updated as alterations are made in the model.

Alternatively, the programmer can define these parameters in a "parametric" manner, which allows the programmer to assign parameter values with respect to a particular target. To illustrate, if the programmer defines the intensity in a "parametric" manner, he specifies that all of the lights which are focused on a particular target or are in a particular color or are mounted on a particular truss, shall have a particular intensity, for example an intensity of 65%. The target or color or truss may be chosen by utilizing a dialogue box to enter the symbolic name that identifies it or by selecting it with the input device as discussed above.

In another variation, the programmer specifies that all of the lights focused on a particular target should be adjusted to produce a desired intensity at the target. The system calculates the required intensity for each light so that for the combined effect of all the lights focused on the target achieves the desired lighting level.

Defining Color

When the programmer is specifying the color for a particular light or group of lights, he may select the color in a number of ways. Once the programmer has selected a light to be defined as part of the model, the particular type of light that has been selected is known. Correspondingly, the color palette appropriate for that type of lighting instrument can be presented to the programmer in a dialogue box. The programmer can use the palette to select the color. In this manner, the programmer can review the available colors and select the desired one. The color palette can be presented in a familiar format such as by the chromaticity triangle. Alternatively, the programmer may wish to enter an alphanumeric name that has been associated with a color he desires. As another possibility, the programmer may define color in an absolute manner by directly controlling the positions of the color filtration elements in the lighting unit. This is obtained by defining the position of the color wheels (for example in a VL2B™), or by defining the angles for each of the three color filters (magenta, amber, blue) of a lighting unit such as in a VL4™.

In a preferred embodiment of the invention, the programmer is able to assign the color of a lighting instrument by establishing "preset colors", similar to the control offered by the Vari-Lite console. Typically the color produced by a lighting instrument can be defined with "standard" colors (assigned by the manufacturer) as well as "custom" and "open" colors (assigned by the programmer). When the color is defined by using "custom", the preset color identifier is stored in the cue. The programmer defines "custom" colors indirectly, by means of dereferenced colors. The color that will be recalled when the cue is initiated is the color that is programmed into the "custom" palette at the time when the preset color is stored. When the color of lighting instruments is defined by means of "open" colors, they will be assigned an absolute color.

The color of a selected light can be specified by a prompt which duplicates the Preset Color controls of a Vari-Lite control console. In that console, the action of pressing a preset color button on a console sets the currently selected lights to the color assigned to that preset color.

Gobo Selection

Similarly, if the lighting instrument being defined achieves beam modification by means of a gobo, the data base can maintain tables of the available gobo selections for each lighting instrument being used and the gobo selected. When selecting the gobo to be used to project a desired beam configuration, the programmer can be presented with a graphical representation containing all of the possibilities. The programmer then merely selects the gobo that he would like to use to achieve the desired effect from the alternatives presented. Alternatively, the programmer can specify his gobo selection by indicating the stop position of the gobo wheel.

When defining other beam characteristics, such as beam size, intensity, and edge the programmer may specify these values by entering the desired values in a dialogue box.

STORING CUE DATA

Once the programmer has obtained the desired lighting effect for a particular cue by manipulating the

entry, or using the spreadsheet mode as discussed above. As the programmer modifies the model, e.g., as he moves an object to a new location on the display screen, the new information is reflected by a change of values in the underlying corresponding data base.

Any needed update to the model can be easily executed in order to incorporate new information into the model to accurately reflect the actual lighting system and stage environment.

UPDATE OF PREVIOUSLY WRITTEN CUE INFORMATION

Furthermore, the parameter data stored as cues may need to be updated if the sequence of events in the performance has been changed, or the location of targets in certain cues is modified. As discussed above, the programmer can update cue data by retrieving the cue from memory by indicating the cue to be retrieved and clicking on the "RCL" icon shown in window 70 of Fig. 19. Thereafter, the previously stored lighting effect as a function of the stored parameters can be displayed on the screen and updated. The programmer can alter any parameter that does not conform to the actual situation. If a target location for the cue has been relocated, the programmer can select the target as discussed above and "drag" it to its new location. As an alternative the programmer may also update any target information by interacting with the modelling system by means of a dialogue box.

DATA TRANSFER

As suggested in Figure 1, the cue data that is written by the programming system to define a lighting effect is translated into data that may be transferred directly to a lighting control console or to local lighting instruments. (In the latter case, the modelling computer is serving as the console.) The syntax of the data should be mutually interpretable by the modelling and programming system disclosed herein and by the conventional control consoles and lighting instruments. In addition to being machine readable, it is preferrable that the data format be human readable as well. A readable textual representation of the data makes the debugging process easier. The modelling and programming tool, and the actual lighting system should be able to read and write in this data format to facilitate bidirectional communication. Additionally, error handling capabilities allow the system to be fault tolerant.

The data format also provides a uniform method for storing all of the cue information that is necessary for each lamp. The file can be arranged in a manner that lists the name and location of each lamp, and stores the parameter data for every cue in the show.

Using the bidirectionality facilitated by this data format, the user can transfer data which has been created or modified in the control console back into the modelling system for subsequent display and further modification.

REAL-TIME RENDERING

The data that has been stored by the modelling system may be processed by a rendering device in order to generate photo-realistic images that represent the stored data. This service is provided for example by the use of a RenderMan program. See "The RenderMan Interface" published by Pixar, 3240 Kerner Blvd., San Rafael, CA 94901. This document details the services available from a RenderMan rendering program.

These images can be utilized to depict what the lighting effect is going to look like. Further, utilizing computer animation techniques to project a continuous sequence of cues at speeds which approach real-time, the viewer can observe chases and the dynamics of the lighting show. The quality of the simulation is a function of the sophistication of the hardware and software platforms that are selected to do the rendering. The sophistication of the image that is available to an average user will improve as the costs and processing power of available systems improve. The technology is moving closer to lower cost systems with excellent simulation capabilities. Ultimately these rendering devices can be used by lighting designers to present a client with a completely simulated performance complete with all of the lighting effects.

Using speaker-independent voice recognition systems with large vocabularies such as those sold by Kurzweil Applied Intelligence of Waltham, MA, the modelling system can provide spoken access to any of its capabilities.

VIRTUAL CONSOLE / VIRTUAL REALITY

With equipment developed by VPL, 656 Bair Island Road, Redwood City, CA, 94063, the 2-D display

for simultaneously generating and displaying views of said lighting scenes from different perspectives.

6. A system as defined in claim 1 in which said means for computing said representations includes means for displaying the contributions of the site, light location and light parameter values to the aggregate lighting scene.

7. A system as defined in claim 1 including means for computing pan and tilt values as a function of lighting instrument location and target position.

8. A system as defined in claim 1 including means for organizing the components of said site parameters on a hierarchical basis.

9. A system as defined in claim 1 including means for generating menus, dialog boxes and icons to facilitate entering of said data.

10. The system according to claim 1 in which said representations constitute cues which can be stored and recalled at will.

11. A system according to claim 1 in which lighting parameters can be specified in terms of target positions.

12. A system according to claim 1 in which said lighting targets include representations of performers and wherein lighting parameter data can be defined and processed to illuminate portions of said performer.

13. A system according to claim 1 including means for representing target volume as an object in the system and wherein said volume can be moved and sized, producing corresponding adjustments in associated lighting parameters.

14. A system according to claim 1 in which said means for entering site data includes means for generating a plurality of orthogonal views on said monitor to display all the positional coordinates of objects in said site.

15. A system according to claim 1 in which the output of said computing system is linked to said console for the exchange of lighting parameters.

16. A system according to claim 1 in which said modelling system includes means for displaying model data in a variety of selectable formats.

17. A system according to claim 1 in which said modelling system includes means for receiving from said console, data describing actual productions for review and modification by said modelling system.

18. A system according to claim 1 in which said modelling system includes means for controlling said lighting instruments from said modelling system.

19. A system according to claim 1 in which said modelling system includes means for specifying the relation between model objects.

20. A system according to claim 1 in which said modelling system includes means for simulating dynamic aspects of said lighting scenes.

21. A method for modelling a lighting production including the production site and the lighting designs produced by the lighting console and lighting instruments, said modelling method employing a computing system having a graphic display program, data entry means, data processing means including a graphic generating program, and a monitor; said method comprising the steps of
(1) entering data into said computing system describing relevant characteristics of said lighting site including lighting targets;
(2) entering data into said computing system identifying the lighting instruments to be involved in a scene;

37. A method according to claim 21 including receiving from said console, data describing actual productions for review and modification by said modelling system.

38. A method according to claim 21 including controlling said lighting instruments from said modelling system

39. A method according to claim 21 including specifying the relation between model objects.

40. A method according to claim 21 including simulating dynamic aspects of said lighting scenes.

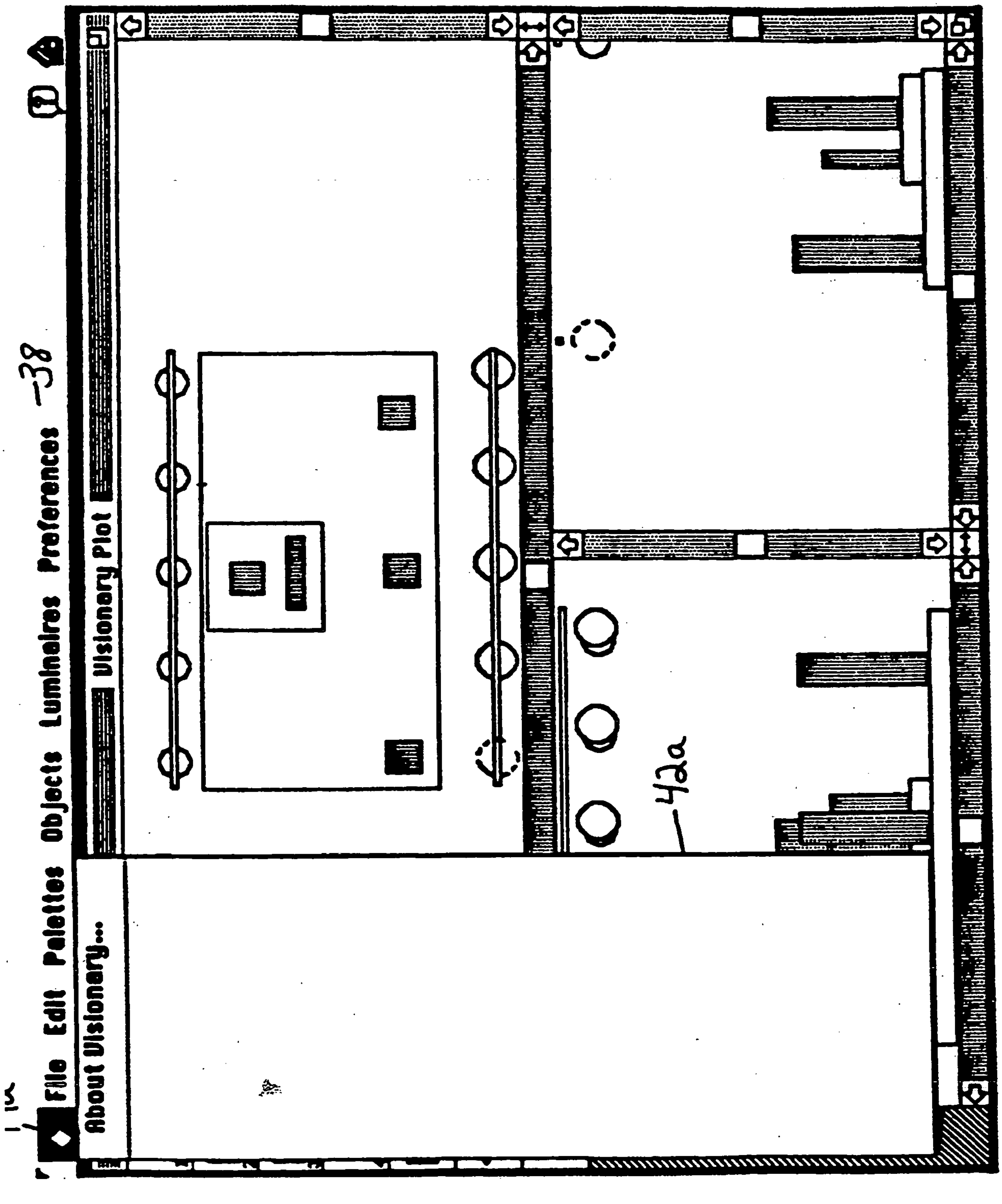

File Edit Palettes Objects Luminaires Preferences
38
About Visionary...
Visionary Plot
42a

FIG. 2a

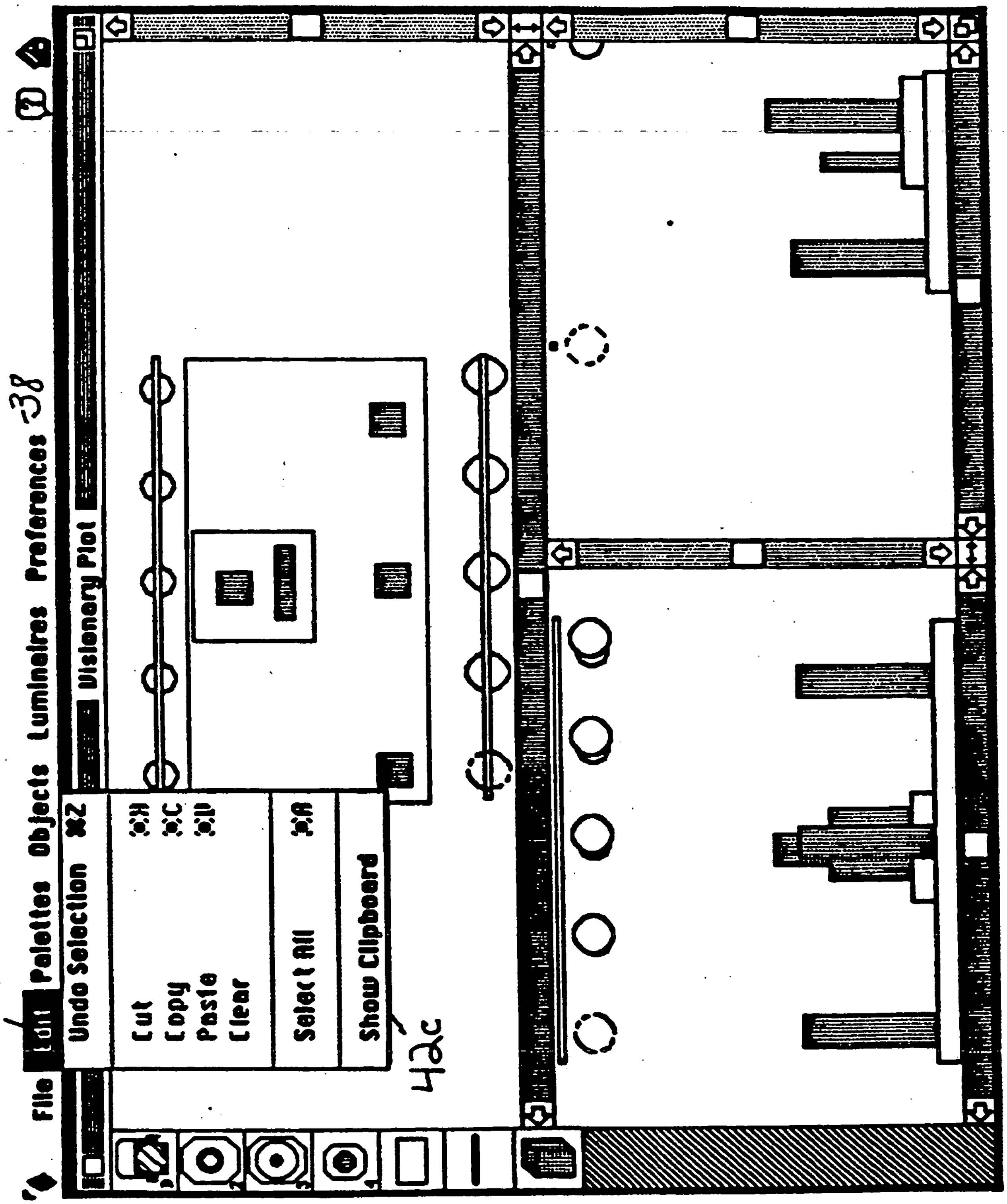
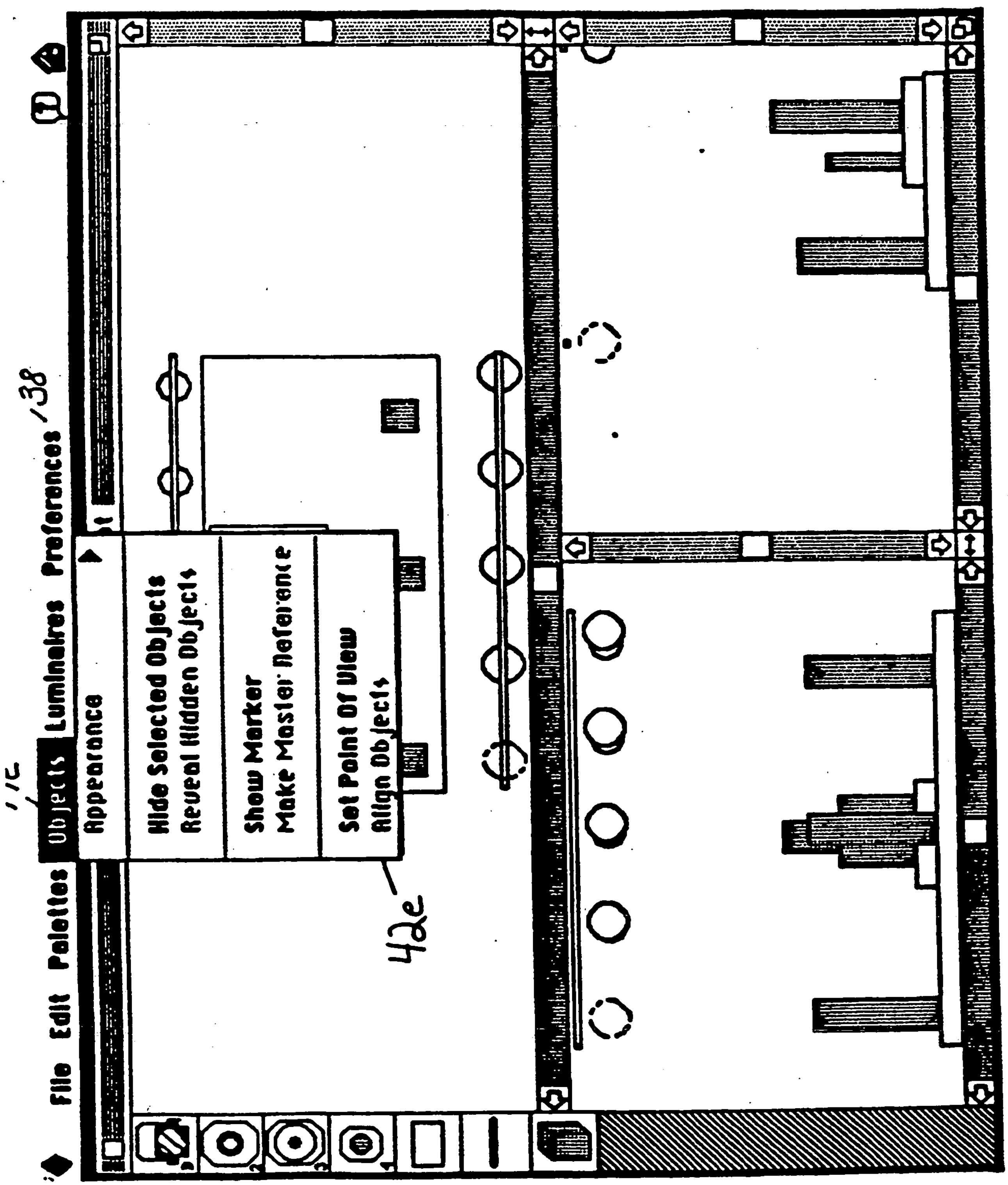

File
Edit
Palettes
Objects
Luminaires
Preferences
38
Undo Selection ⌘Z
Cut ⌘X
Copy ⌘C
Paste ⌘V
Clear
Select All ⌘A
Show Clipboard
42c
Visionary Plot

FIG. 2c



File
Edit
Palettes
Objects
Luminaires
Preferences
38
Appearance
Hide Selected Objects
Reveal Hidden Objects
Show Marker
Make Master Reference
Set Point Of View
Align Objects
42e

FIG. 2e

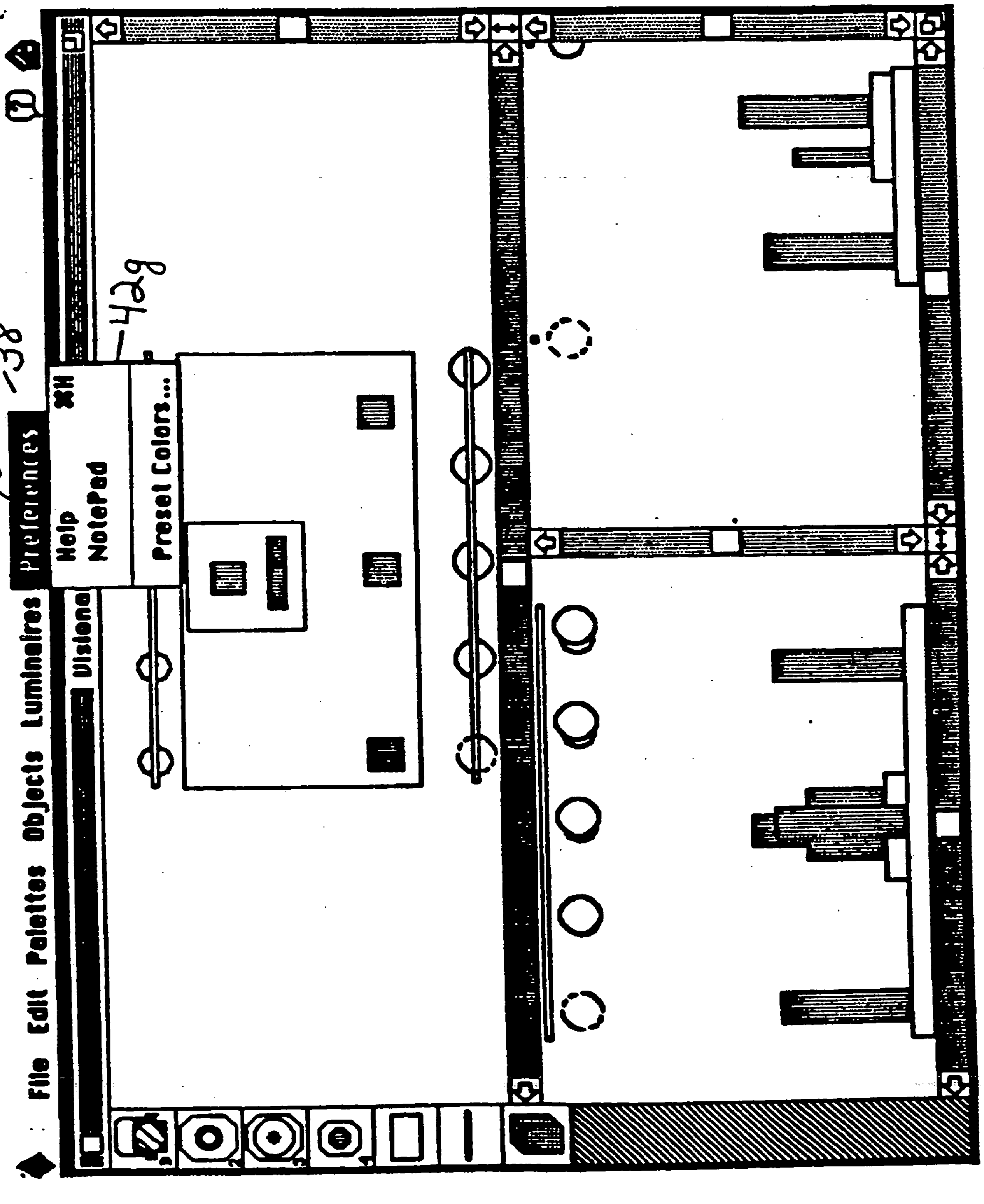

File
Edit
Palettes
Objects
Luminaires
Visions
Preferences
Help
NotePad
⌘H
Preset Colors...
38
42g

FIG. 2g

Model Window Sample (English Units)

Name	Type	Chan. No.	Shown/ Hidden	Surface Color	Focus	Color	Beam Size	Support	Dimensions	Location from Master Marker or Support	Rotation about Support
500	Conv.	500	Yes	-	Drummer	color	10	Truss 2-Up	-	Upstage 10' 0" Above 30' 0" Stage Left 10' 10"	Upstage 0 Vertical 0 Cross-stage 0
501	Conv.	501	Yes	-	Singer	Roscolux #02	15	Truss 2-Down	-	Upstage 10' 0" Above 30' 0" Stage Right 10' 10"	Upstage 0 Vertical 0 Cross-stage 0
502	Conv.	502	Yes	-	Singer	GamColor #520	20	Truss 2-up	-	Upstage 10' 0" Above 30' 0" Stage Left 0' 0"	Upstage 0 Vertical 0 Cross-stage 0
Stage	Setpiece	-	Yes	color	-	-	-	Floor/Ceiling	Height 12" Depth 120" Width 240"	Upstage 0' 0" Above 0' 0" Stage Left 0' 0"	Upstage 0 Vertical 0 Cross-stage 0
Drummer	Target	-	Yes	color	-	-	-	Stage	Height 60" Depth 48" Width 120"	Upstage 5' 0" Above 3' 0" Stage Left 0' 0"	Upstage 0 Vertical 0 Cross-stage 0
Singer	Target	-	Yes	color	-	-	-	Stage	Heigth 66" Depth 18" Width 18"	Downstage 2' 0" Above 0' 0" Stage Left 0' 0"	Upstage 0 Vertical 0 Cross-stage 0
Truss 1	Truss	-	Yes	color	-	-	-	Floor/Ceiling	Length 240" Width 24"	Upstage 30' 0" Above 30' 0" Stage Left 0' 0"	Upstage 0 Vertical 0 Cross-stage 0
Truss 2	Truss	-	Yes	color	-	-	-	Floor/Ceiling	Length 240" Width 24"	Upstage 10' 0" Above 30' 0" Stage Left 0' 0"	Upstage 0 Vertic Cross-stage 0
Pipe 1	Truss	-	Yes	color	-	-	-	Floor/Ceiling	Length 240"	Downstage 9' 10" Below 6' 7" Stage Left 0' 0"	Upstage 0 Vertical 0 Cross-stage 0
Pipe 2	Truss	-	No	color	-	-	-	Floor/Ceiling	Length 240"	Downstage 6' 7" Below 6' 7" Stage Left 0' 0"	Upstage 0 Vertical 0 Cross-stage 0

FIG. 4a

Type	Name	Channel Number	Shown/ Hidden	Surface Color	Focus	Color	Beam Size	Support	Dimensions
VL2B	Name	1..499	Yes or No	.	.	.	.	Support Name	.
VL4	Name	1..499	Yes or No	.	.	.	.	Support Name	.
Conv.	Name	500..999	Yes or No	.	Target Name	(Your gel color here)	size	Support Name	.
						(Your gel number here)			
Truss	Truss Name	.	Yes or No	(Your surface color here)	.	.	.	Support Name	Height 0..xxx' 0..11" or 0.00..xx.99 m Depth 0..xxx' 0..11" or 0.00..xx.99 m Width 0..xxx' 0..11" or 0.00..xx.99 m
Pipe	Pipe Name	.	Yes or No	(Your surface color here)	.	.	.	Support Name	Height 0..xxx' 0..11" or 0.00..xx.99 m Depth 0..xxx' 0..11" or 0.00..xx.99 m Width 0..xxx' 0..11" or 0.00..xx.99 m
Setpiece	Setpiece Name	.	Yes or No	(Your surface color here)	.	.	.	Support Name	Height 0..xxx' 0..11" or 0.00..xx.99 m Depth 0..xxx' 0..11" or 0.00..xx.99 m Width 0..xxx' 0..11" or 0.00..xx.99 m
Target	Target Name	.	Yes or No	(Your surface color here)	.	.	.	Support Name	Height 0..xxx' 0..11" or 0.00..xx.99 m Depth 0..xxx' 0..11" or 0.00..xx.99 m Width 0..xxx' 0..11" or 0.00..xx.99 m

FIG. 5a

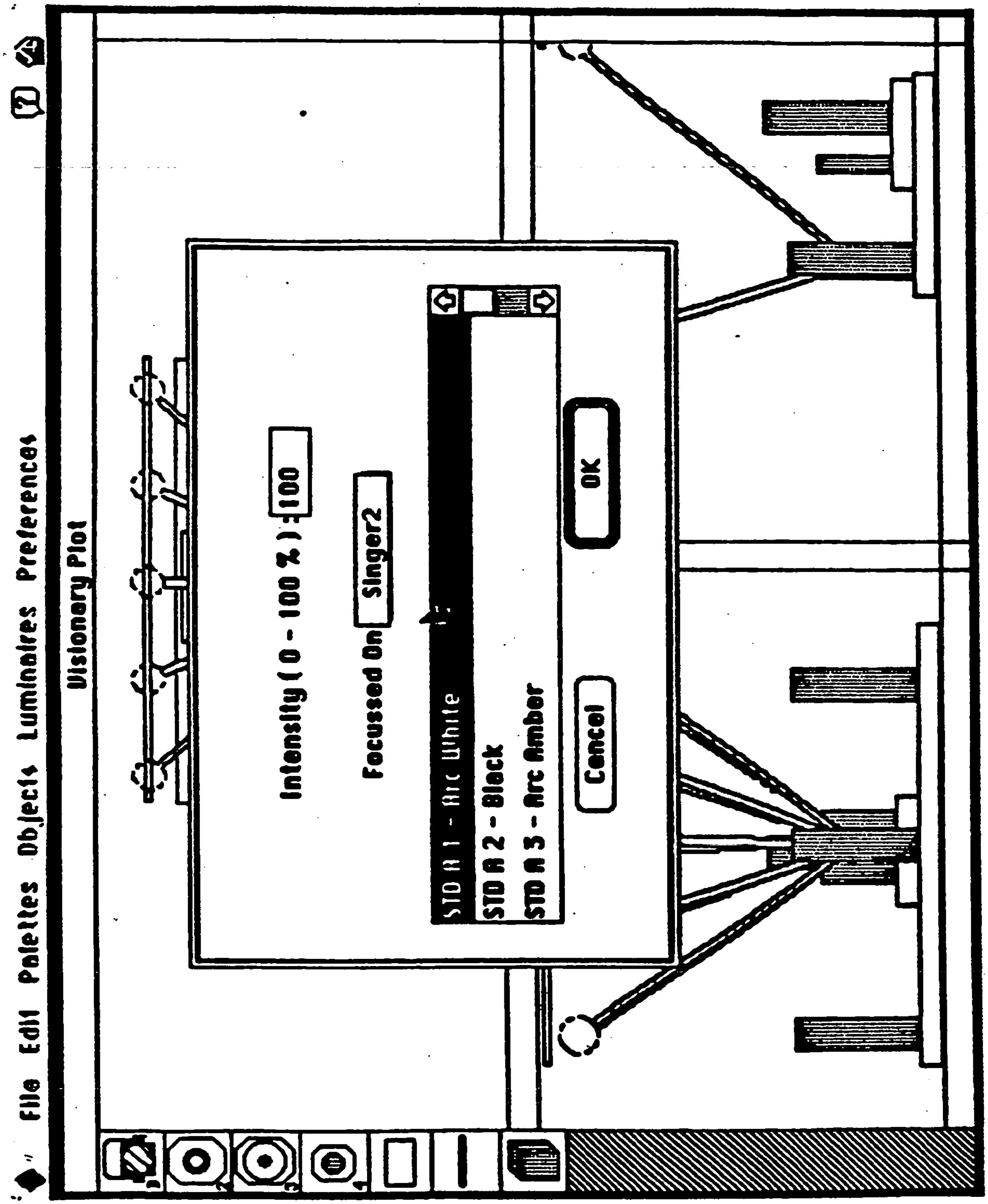

File Edit Palettes Objects Luminaires Preferences
Visionary Plot
Intensity (0 - 100 %) :
100
Focussed On
Singer2
STD A 1 - Arc White
STD A 2 - Black
STD A 3 - Arc Amber
OK
Cancel

FIG. 6

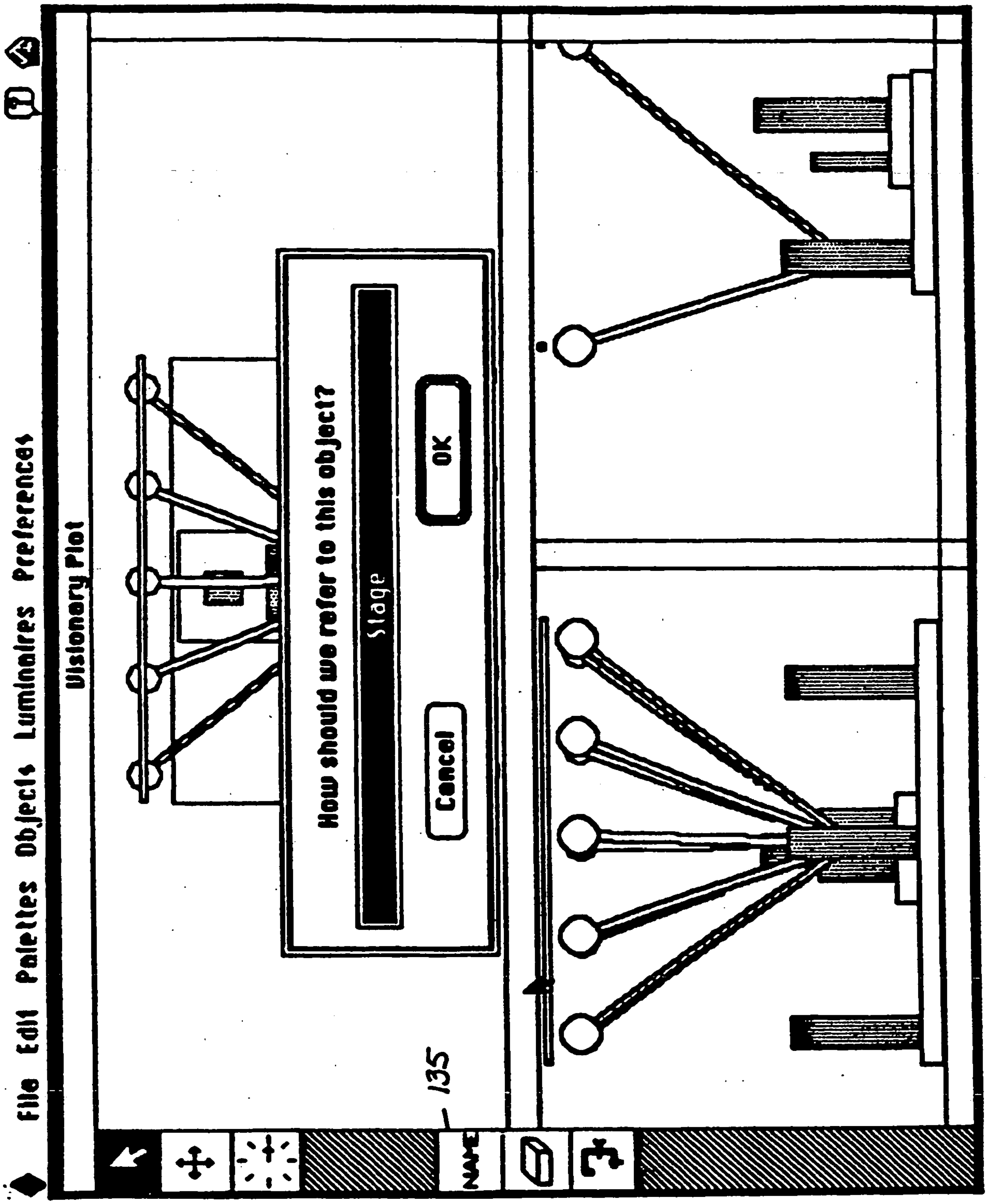

File Edit Palettes Objects Luminaires Preferences
Visionary Plot
How should we refer to this object?
Stage
OK
Cancel
NAME
135

FIG. 8

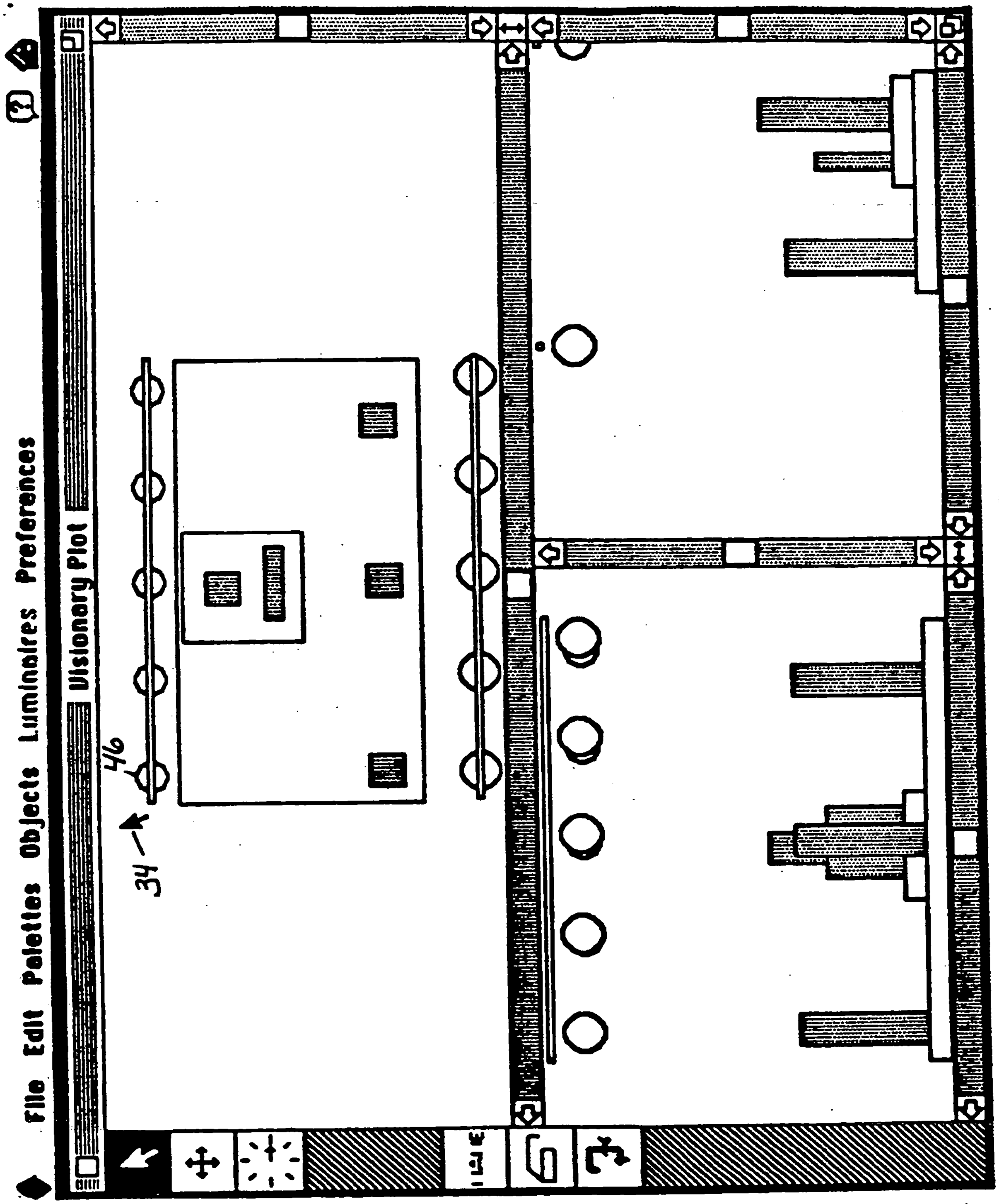

File Edit Palettes Objects Luminaires Preferences
Visionary Plot
34
46

FIG. 10a

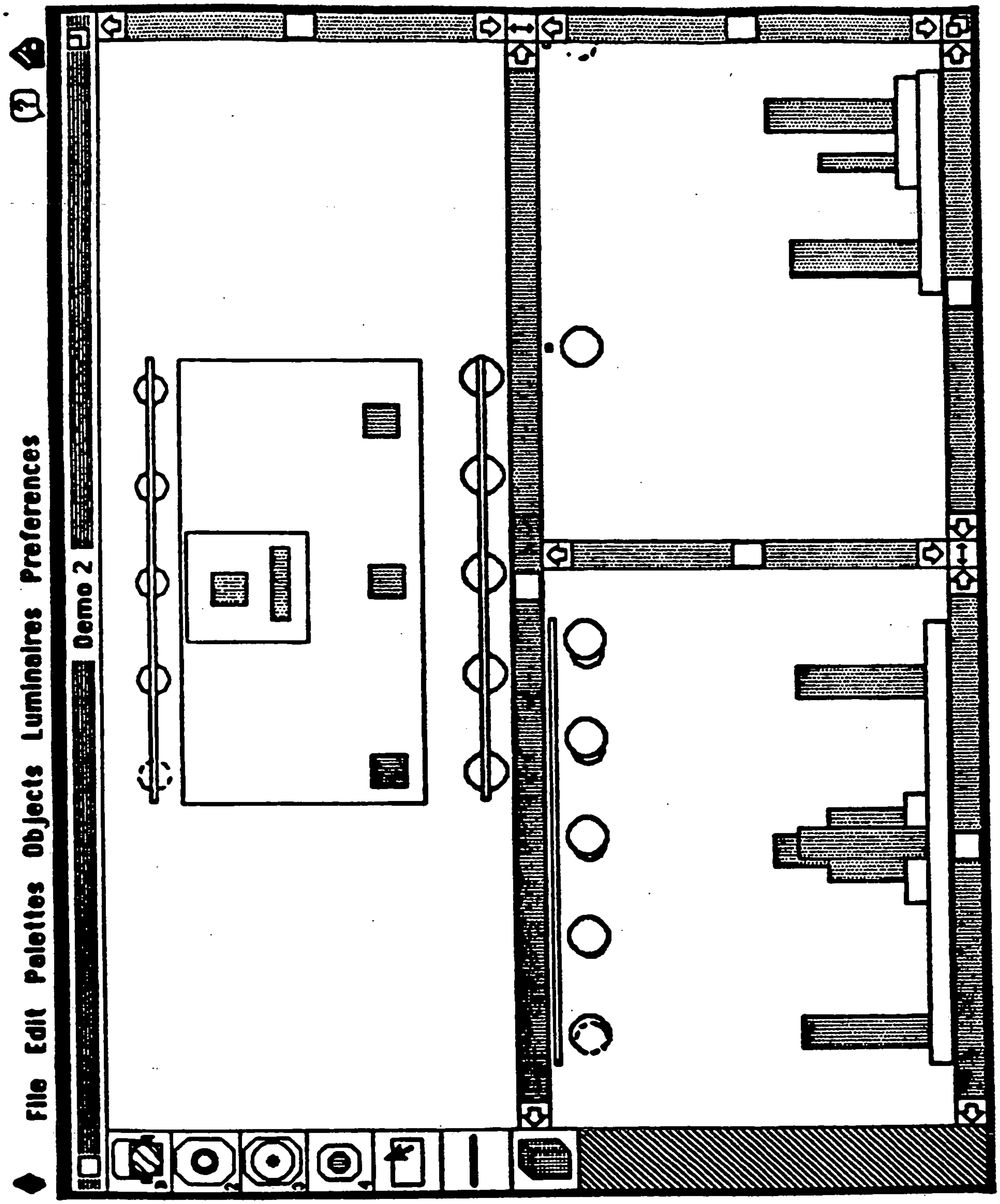


FIG. 10c

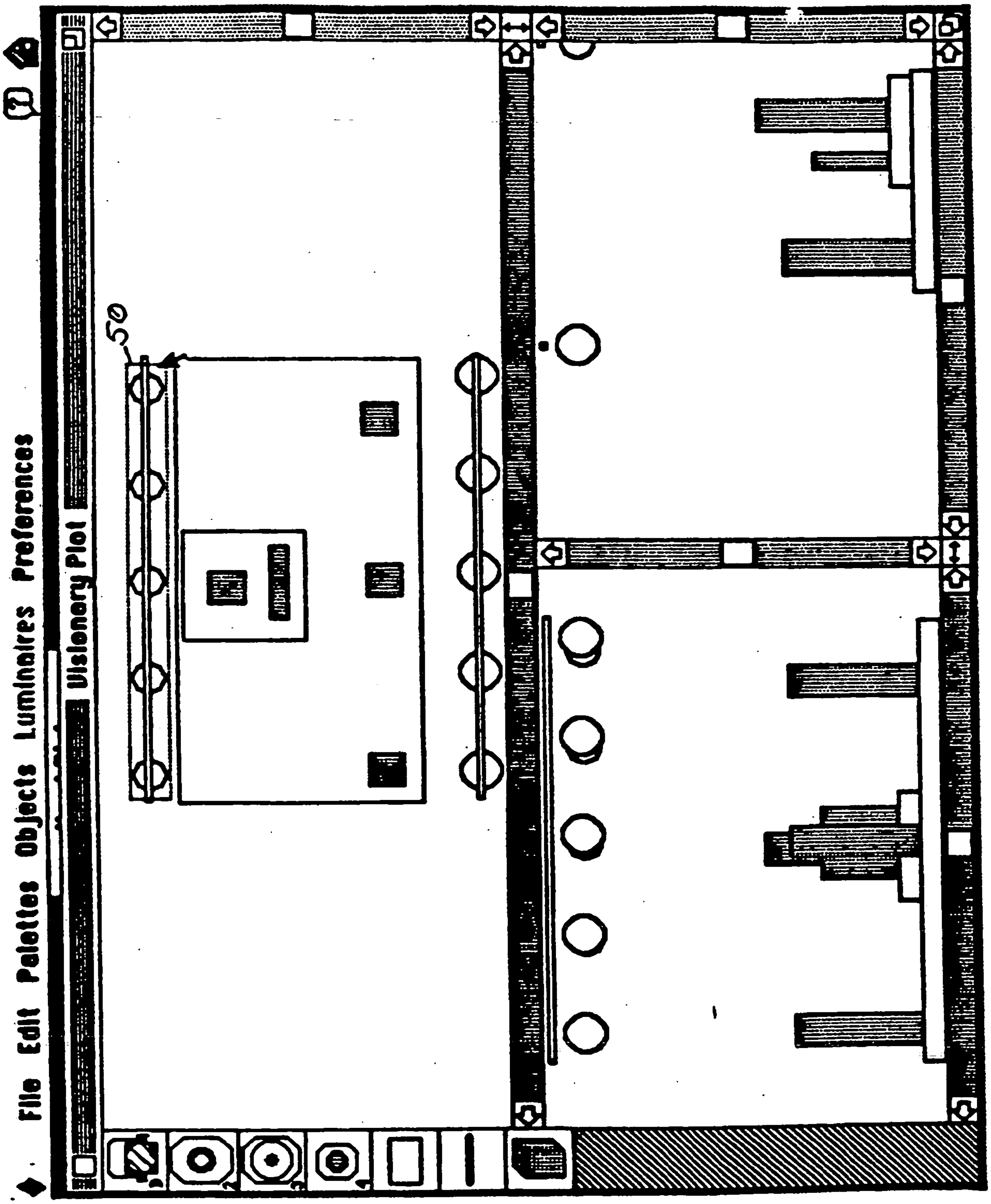

File Edit Palettes Objects Luminaires Preferences
Visionary Plot
50

FIG. 11b

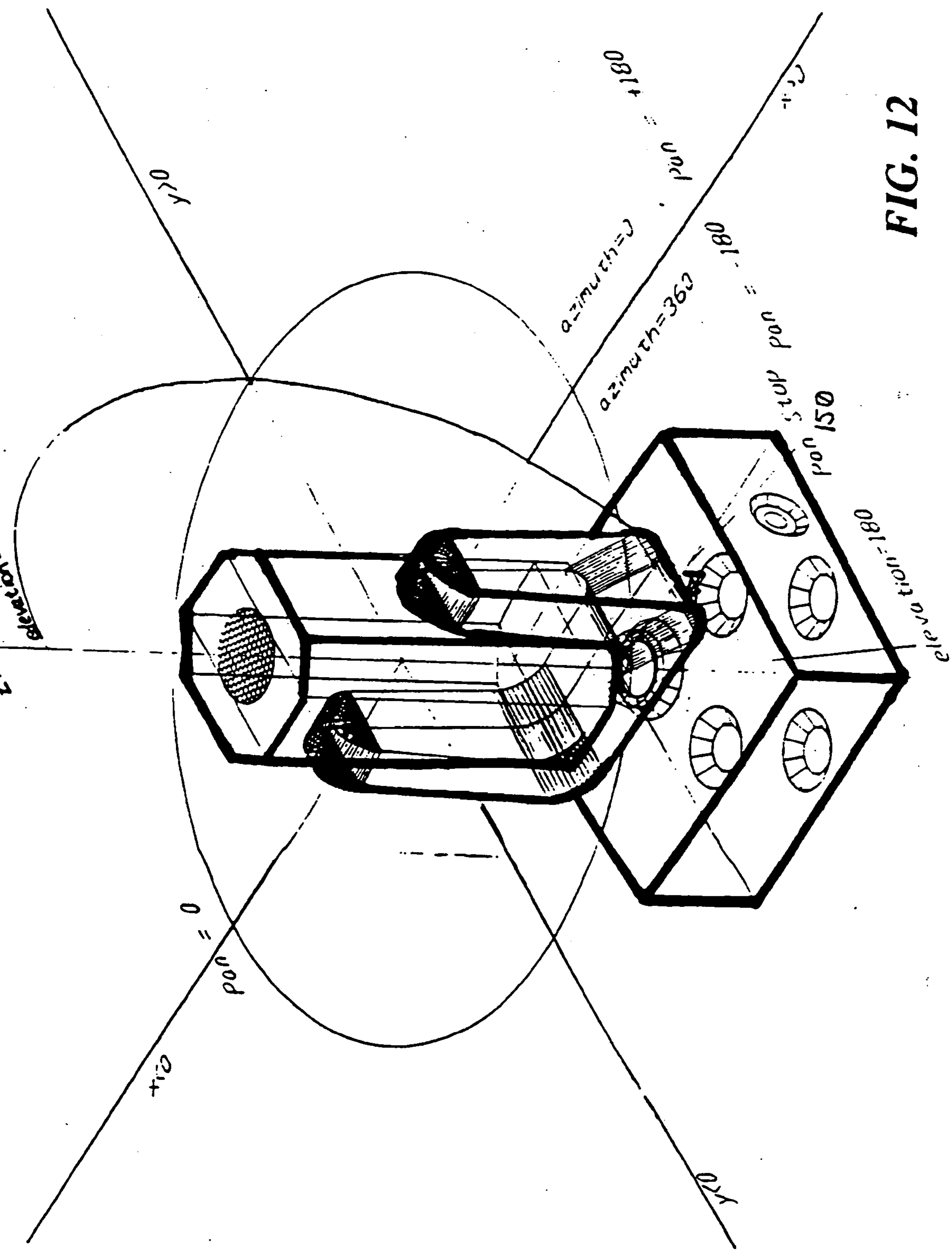
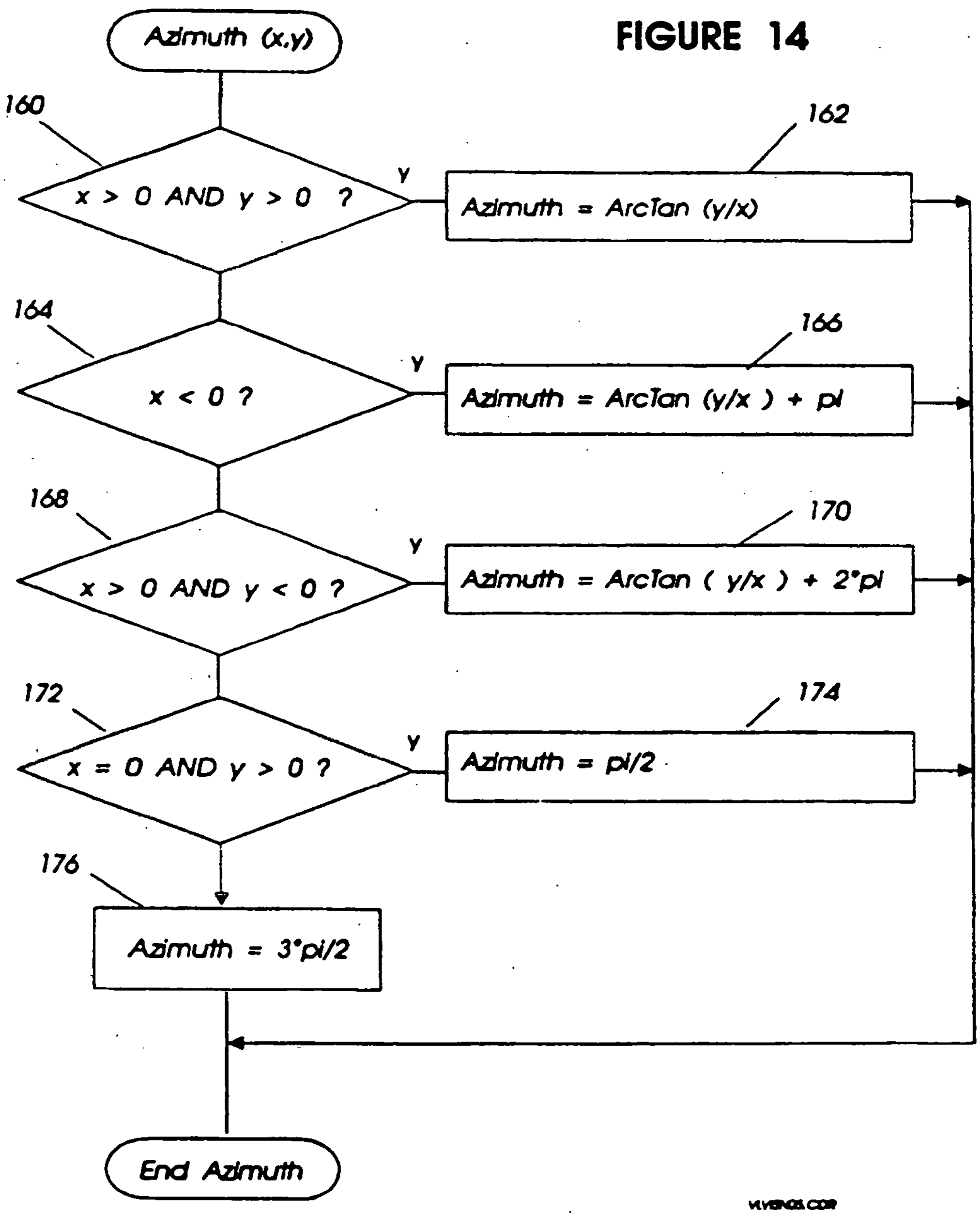


FIG. 12

Azimuth

FIGURE 14



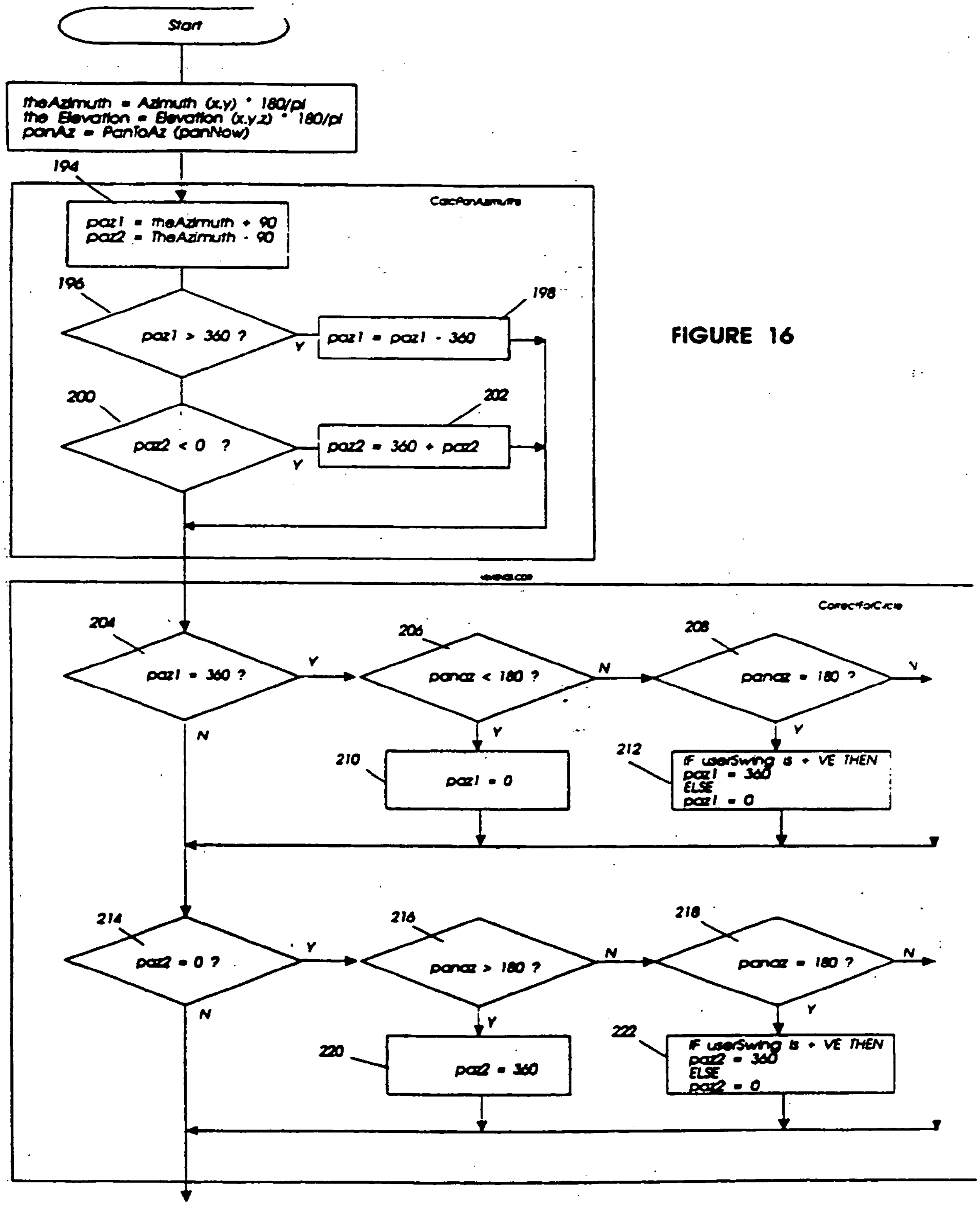

Start
theAzimuth = Azimuth (x,y) * 180/pi
the Elevation = Elevation (x,y,z) * 180/pi
panAz = PanToAz (panNow)
194
CalcPanAzimuths
paz1 = theAzimuth + 90
paz2 = TheAzimuth - 90
196
paz1 > 360 ?
Y
198
paz1 = paz1 - 360
200
paz2 < 0 ?
Y
202
paz2 = 360 + paz2
CorrectForCircle
204
paz1 = 360 ?
Y
N
206
panaz < 180 ?
N
Y
208
panaz = 180 ?
Y
210
paz1 = 0
212
IF userSwing is + VE THEN
paz1 = 360
ELSE
paz1 = 0
214
paz2 = 0 ?
Y
N
216
panaz > 180 ?
N
Y
218
panaz = 180 ?
N
Y
220
paz2 = 360
222
IF userSwing is + VE THEN
paz2 = 360
ELSE
paz2 = 0

FIGURE 16

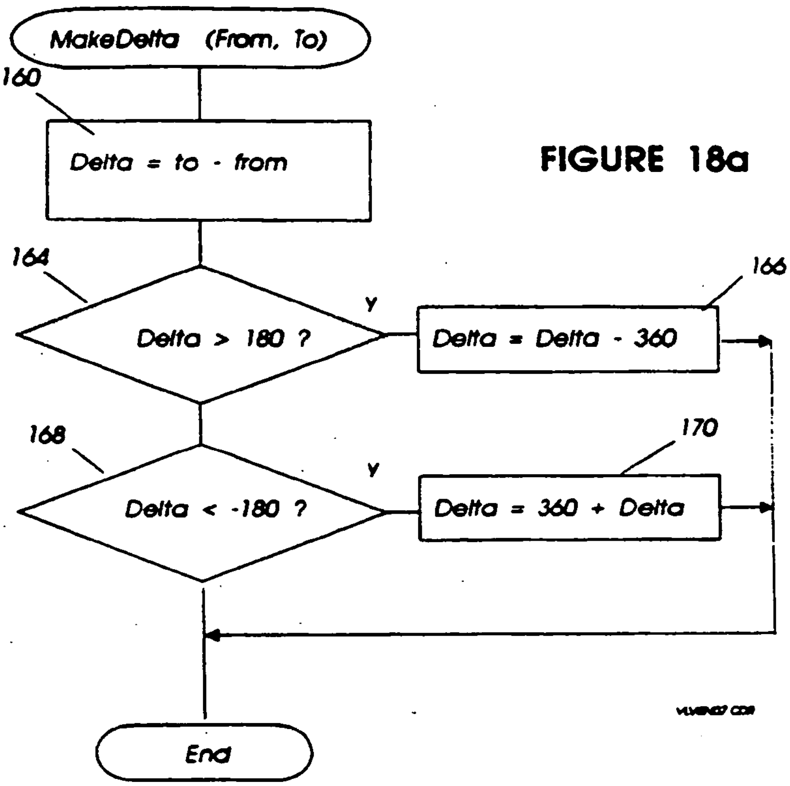


FIGURE 18a

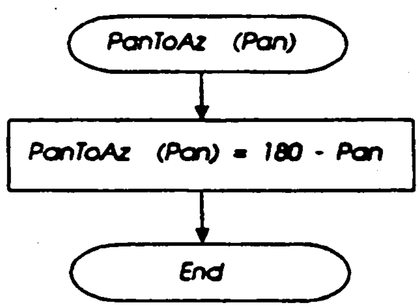


FIGURE 18b

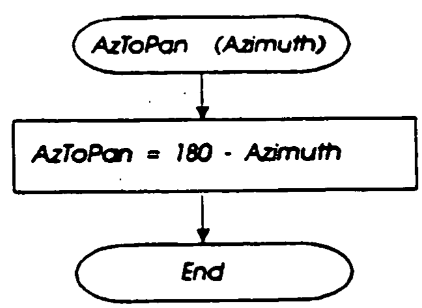


FIGURE 18c

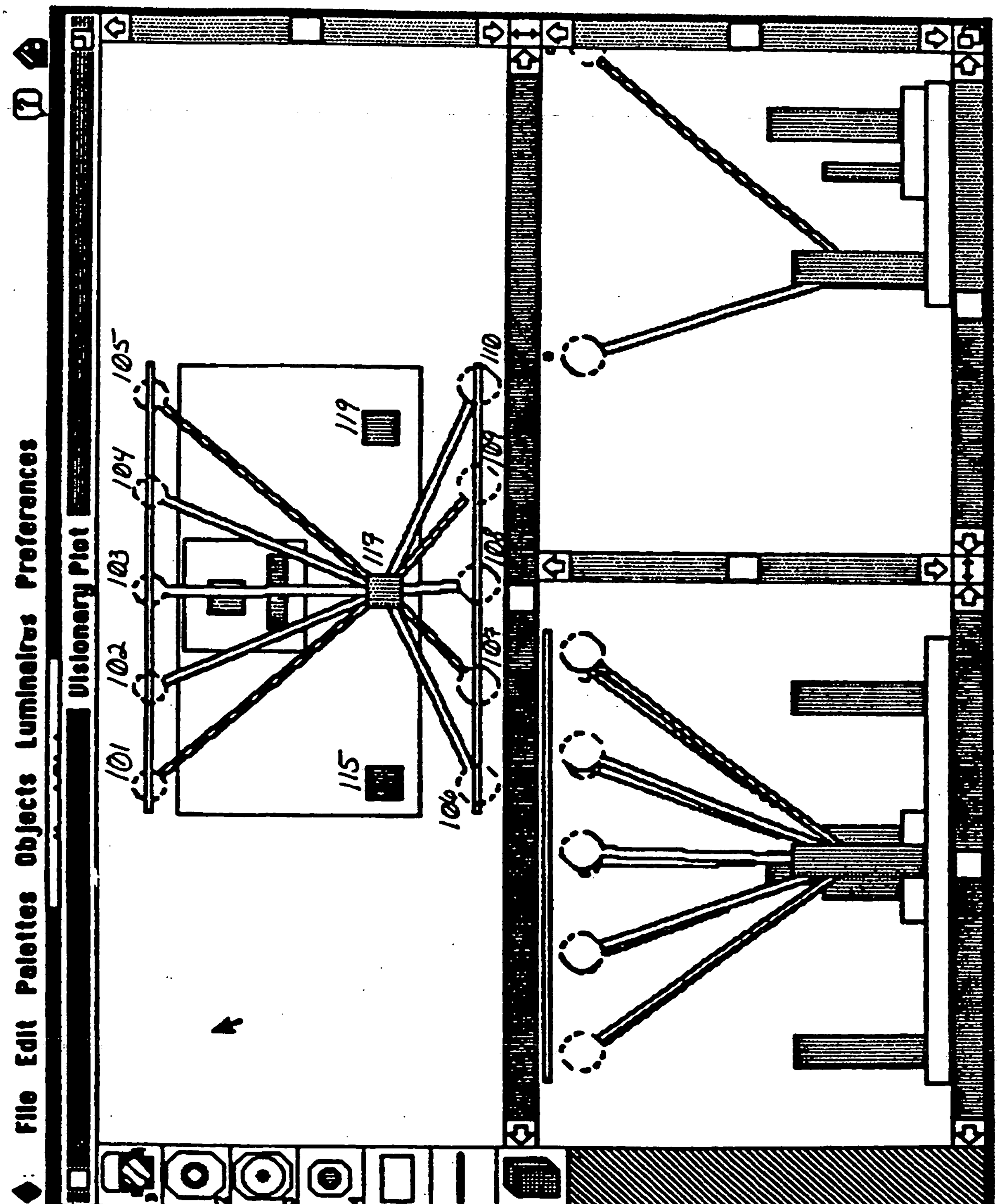

File Edit Palettes Objects Luminaires Preferences
Visionary Plot
101
102
103
104
105
106
107
108
109
110
115
117
119

FIG. 20